



Fundamentos de Python

Lección 2: Posición estratégica de Python. Preparando el entorno de desarrollo.

ÍNDICE

Posición estratégica del lenguaje Python.	2
1 PRESENTACIÓN Y OBJETIVOS.....	2
2 Posición estratégica del lenguaje Python.....	3
3 Preparando el entorno de desarrollo: Cómo instalar python en los diferentes sistemas operativos.....	7
Instalación de Python en Ubuntu 16.04	7
Gestor de paquetes Pip en linux	14
Instalación de Python en Windows 10.....	15
Gestor de paquetes Pip en Windows	17
4 Puntos clave.....	19

Posición estratégica del lenguaje Python.

1 PRESENTACIÓN Y OBJETIVOS

En este capítulo haremos un breve repaso sobre la posición estratégica del lenguaje Python, destacando las áreas donde es aplicado.

Por otro lado, prepararemos el entorno de desarrollo en nuestras máquinas, para poder instalar tanto el lenguaje Python como su intérprete, necesario para poder ejecutar su código. Para facilitar la tarea del alumnado, se mostrará una guía de instalación de Python para los sistemas operativos: Windows 10 y Ubuntu 16.04.



Objetivos

- **Conocer la posición estratégica de Python.**
- **Conocer sus principales áreas de aplicación.**
- **Saber instalar Python en cualquiera de los sistemas operativos más utilizados en la actualidad.**

2 POSICIÓN ESTRATÉGICA DEL LENGUAJE PYTHON

El lenguaje de programación Python ha logrado ser el centro de atención de muchos desarrolladores de software, debido a su alta interpretabilidad y facilidad de uso. Además, como comentamos en el capítulo anterior, debido a que Python es código abierto, se ha podido liberar una gran cantidad de módulos y librerías diseñados para resolver problemas de diferente índole.

Existen varios mecanismos para medir la popularidad de los lenguajes de programación, como el índice TIOBE o PYPL. Mientras que TIOBE calcula la popularidad de un lenguaje de programación atendiendo a la cantidad de sitios indexados en Google, Yahoo! y Bing que mencionan un lenguaje; PYPL mide la popularidad de los lenguajes de programación teniendo en cuenta con qué frecuencia los desarrolladores de software buscan tutoriales en Google sobre un lenguaje de programación. Es decir, a mayor búsqueda de información sobre cómo utilizar una tecnología de un lenguaje de programación, mayor popularidad tiene este lenguaje.

Teniendo en cuenta la clasificación realizada por el índice PYPL (del inglés *PopularitY of Programming Language Index*), Python se ha consolidado como el lenguaje de programación más popular en el año 2020, así como en 2021 (a fecha de realización de esta memoria).

Worldwide, Feb 2021 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	30.06 %	+0.3 %
2		Java	16.88 %	-1.7 %
3		JavaScript	8.43 %	+0.4 %
4		C#	6.69 %	-0.6 %
5	↑	C/C++	6.5 %	+0.5 %

Figura 2.1: Ranking PYPL 2021

Cabe destacar que Python no siempre ha sido considerado como el lenguaje de programación más utilizado o famoso, si no que ha necesitado una larga trayectoria para ganarse este puesto. Fíjese en la siguiente gráfica, donde el lector podrá observar como desde el año 2005 Python ha tomado una tendencia creciente, hasta que a mediados del año 2018 consiguió convertirse en el lenguaje de programación más popular

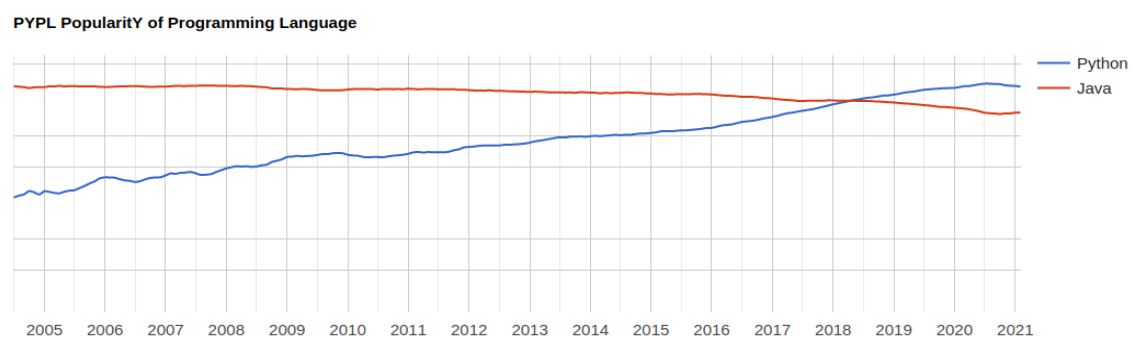


Figura 22: Evolución de la popularidad de Python

Por otro lado, si consideramos el índice de popularidad calculado por TIOBE, podemos comprobar que Python aún no se ha ganado el primer puesto de este ranking. Aun así, como se muestra en la siguiente imagen, se puede comprobar que aunque se encuentre en tercera posición de este ranking, su tendencia es creciente; mientras que algunos de sus rivales como Java tienen una tendencia decreciente.

Feb 2021	Feb 2020	Change	Programming Language	Ratings	Change
1	2	▲	C	16.34%	-0.43%
2	1	▼	Java	11.29%	-6.07%
3	3		Python	10.86%	+1.52%
4	4		C++	6.88%	+0.71%
5	5		C#	4.44%	-1.48%
6	6		Visual Basic	4.33%	-1.53%

Figura 2.3: Ranking TIOBE

El hecho de que Python haya ido ganando adeptos en los últimos años puede deberse, entre otros, a su alta versatilidad, ya que puede ser utilizado en diferentes disciplinas. Entre las disciplinas donde podemos utilizar Python, queremos destacar tres de las que consideramos esenciales ya que día a día más empresas buscan más expertos en este área. Hablamos, por supuesto, de las especialidades de **Machine Learning, Big Data y Hacking**.

En relación a la especialidad de Machine Learning, podemos encontrar una gran variedad de librerías que permiten implementar de una forma sencilla técnicas avanzadas como redes neuronales, regresiones o técnicas de clustering, entre otros. Además, un aspecto crucial a la hora de resolver problemas de Machine Learning es la capacidad de poder representar gráficamente los resultados obtenidos. Este tipo de representaciones son de utilidad para el experto a la hora de interpretar una solución y tomar una serie de decisiones. Algunas de las librerías de Python que pueden resultar de interés para resolver problemas de Machine Learning y visualización son las siguientes:

- | Scikit-learn
- | Matplotlib
- | TensorFlow
- | Keras
- | NLTK
- | Pydoop



Presta atención

Aunque existe una gran variedad de librerías para resolver este tipo de tareas, no hay ninguna regla que nos indique cuál es mejor. Es tarea del experto desarrollador utilizar la librería que mejor se adapte a su situación.

Por último, aunque a lo largo de este máster estudiaréis en detalle sobre cómo utilizar Python para resolver problemas relacionados con hacking, a continuación mostraremos algunas de las herramientas más conocidas que son utilizadas en Python para este propósito:

- | SCAPY
- | IMPACKET
- | Requests
- | Cryptography

Tras este repaso sobre la importancia y posición estratégica de Python en el mercado actual, procederemos a instalar Python en nuestras máquinas para comenzar a utilizarlo. En primer lugar debemos preparar nuestro entorno de desarrollo, y posteriormente procederemos a instalar una versión estable de Python.

3 PREPARANDO EL ENTORNO DE DESARROLLO: CÓMO INSTALAR PYTHON EN LOS DIFERENTES SISTEMAS OPERATIVOS.

En esta sección veremos cómo instalar python en nuestras máquinas. En concreto, mostraremos los pasos a seguir para instalar python en el sistema operativo Linux (Ubuntu 16.04) y Windows 10. Para esta asignatura, así como en el resto de las de este máster, haremos uso de la versión 3.8.7 de Python.

Instalación de Python en Ubuntu 16.04

Debido a la gran importancia e influencia que ha tenido Python en programación, la mayoría de las distribuciones de Ubuntu incluyen una versión preinstalada, aunque suele ser antigua. Para conocer la versión instalada por defecto ejecutaremos en la terminal el siguiente comando:

```
python -V
```

En una instalación básica de Ubuntu 16.04 obtendremos la siguiente respuesta:

```
Python 2.7.12
```




Presta atención

En algunos sistemas podréis encontrar instalada la versión 2 o 3.5 de Python. En estos casos, debéis seguir este manual para instalar la versión 3.8.7. Si viniera instalada por defecto, podéis pasar al siguiente apartado.

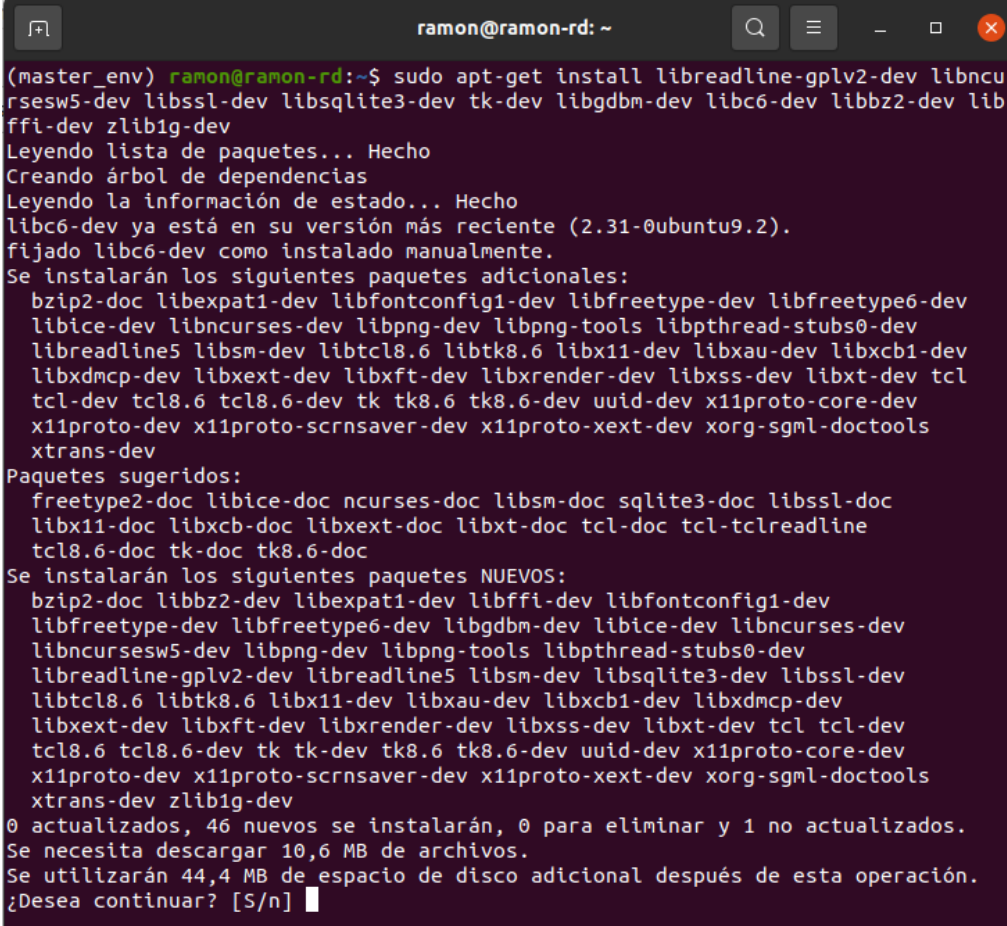
Sin embargo, como hemos comentado, en esta asignatura trabajaremos con la última versión estable de Python (a fecha de realización de este manual), la versión 3.8.7. Para instalarlo, necesitamos instalar previamente una serie de paquetes que son necesarios para que funcione esta versión:

Paso 1: actualizamos los paquetes con la orden “sudo apt-get update” y posteriormente instalamos los paquetes “build-essential” y “checkinstall” como se muestra a continuación:

```
ramon@ramon-rd: ~  
(master_env) ramon@ramon-rd:~$ sudo apt-get install build-essential checkinstall  
Leyendo lista de paquetes... Hecho  
Creando árbol de dependencias  
Leyendo la información de estado... Hecho  
build-essential ya está en su versión más reciente (12.8ubuntu1.1).  
Fijado build-essential como instalado manualmente.  
Paquetes sugeridos:  
  gettext  
Se instalarán los siguientes paquetes NUEVOS:  
  checkinstall  
0 actualizados, 1 nuevos se instalarán, 0 para eliminar y 1 no actualizados.  
Se necesita descargar 99,3 kB de archivos.  
Se utilizarán 442 kB de espacio de disco adicional después de esta operación.  
¿Desea continuar? [S/n]
```

Figura 3.4: Instalación de paquetes necesarios para Python 1/2

Paso 2: Instalamos los paquetes restantes que son necesarios para que Python pueda funcionar:



```

ramon@ramon-rd: ~
(master_env) ramon@ramon-rd:~$ sudo apt-get install libreadline-gplv2-dev libncu
rsesw5-dev libssl-dev libsqlite3-dev tk-dev libgdbm-dev libc6-dev libbz2-dev lib
ffi-dev zlib1g-dev
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
libc6-dev ya está en su versión más reciente (2.31-0ubuntu9.2).
fijado libc6-dev como instalado manualmente.
Se instalarán los siguientes paquetes adicionales:
  bzip2-doc libexpat1-dev libfontconfig1-dev libfreetype-dev libfreetype6-dev
  libice-dev libncurses-dev libpng-dev libpng-tools libpthread-stubs0-dev
  libreadline5 libsm-dev libtcl8.6 libtk8.6 libx11-dev libxau-dev libxcb1-dev
  libxdmcp-dev libxext-dev libxft-dev libxrender-dev libxss-dev libxt-dev tcl
  tcl-dev tcl8.6 tcl8.6-dev tk tk8.6 tk8.6-dev uuid-dev x11proto-core-dev
  x11proto-dev x11proto-headers-dev x11proto-xext-dev xorg-sgml-doctools
  xtrans-dev
Paquetes sugeridos:
  freetype2-doc libice-doc ncurses-doc libsm-doc sqlite3-doc libssl-doc
  libx11-doc libxcb-doc libxext-doc libxt-doc tcl-doc tcl-tclreadline
  tcl8.6-doc tk-doc tk8.6-doc
Se instalarán los siguientes paquetes NUEVOS:
  bzip2-doc libbz2-dev libexpat1-dev libffi-dev libfontconfig1-dev
  libfreetype-dev libfreetype6-dev libgdbm-dev libice-dev libncurses-dev
  libncursesw5-dev libpng-dev libpng-tools libpthread-stubs0-dev
  libreadline-gplv2-dev libreadline5 libsm-dev libsqlite3-dev libssl-dev
  libtcl8.6 libtk8.6 libx11-dev libxau-dev libxcb1-dev libxdmcp-dev
  libxext-dev libxft-dev libxrender-dev libxss-dev libxt-dev tcl tcl-dev
  tcl8.6 tcl8.6-dev tk tk-dev tk8.6 tk8.6-dev uuid-dev x11proto-core-dev
  x11proto-dev x11proto-headers-dev x11proto-xext-dev xorg-sgml-doctools
  xtrans-dev zlib1g-dev
0 actualizados, 46 nuevos se instalarán, 0 para eliminar y 1 no actualizados.
Se necesita descargar 10,6 MB de archivos.
Se utilizarán 44,4 MB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]

```

Figura 3.5: Instalación de paquetes necesarios para Python 2/2

Paso 3: A continuación, nos descargamos la versión 3.8.7 desde la página oficial de Python. Podemos hacerlo manualmente (accediendo a su web y realizando la descarga) o mediante comandos en una terminal. Si optamos por la opción de descargarlo por medio de la terminal, realizaremos lo siguiente:

```
ramon@ramon-rd: /opt
(master_env) ramon@ramon-rd:~$ cd /opt
(master_env) ramon@ramon-rd:/opt$ sudo wget https://www.python.org/ftp/python/3.8.7/Python-3.8.7.tgz
--2021-02-11 20:15:14-- https://www.python.org/ftp/python/3.8.7/Python-3.8.7.tgz
Resolviendo www.python.org (www.python.org)... 151.101.132.223, 2a04:4e42:1f::223
Conectando con www.python.org (www.python.org)[151.101.132.223]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 24468684 (23M) [application/octet-stream]
Guardando como: "Python-3.8.7.tgz"

Python-3.8.7.tgz      100%[=====] 23,33M  8,78MB/s   en 2,7s
2021-02-11 20:15:17 (8,78 MB/s) - "Python-3.8.7.tgz" guardado [24468684/24468684]

(master_env) ramon@ramon-rd:/opt$
```

Figura 3.6: Descarga de Python 3.8.7 por terminal



Presta atención

Nótese que hemos creado un directorio temporal al que hemos llamado `opt`, para descargar y almacenar la versión de Python.

Para crear un directorio desde la terminal puede ejecutar la orden `mkdir nombre_directorio`. En nuestro caso, hemos ejecutado la orden:

```
mkdir opt
```

Una vez creado el directorio, accedemos al mismo por medio de la orden `cd`. Posteriormente, con el comando `wget` accedemos y descargamos la versión de Python.

Por otro lado, podemos realizar este mismo proceso descargando Python de su página oficial:

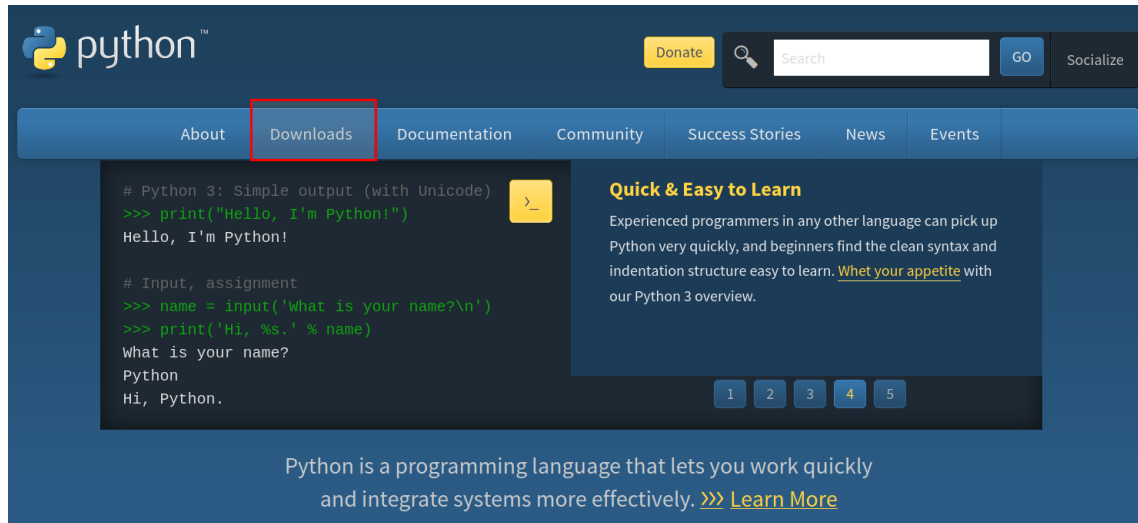


Figura 3.7: Descarga de Python 3.8.7 vía web 1/2

Release version	Release date	Click for more	
Python 3.8.7	Dec. 21, 2020	Download	Release Notes
Python 3.9.1	Dec. 7, 2020	Download	Release Notes
Python 3.9.0	Oct. 5, 2020	Download	Release Notes
Python 3.8.6	Sept. 24, 2020	Download	Release Notes
Python 3.5.10	Sept. 5, 2020	Download	Release Notes
Python 3.7.9	Aug. 17, 2020	Download	Release Notes
Python 3.6.12	Aug. 17, 2020	Download	Release Notes

Figura 3.8: Descarga de Python 3.8.7 vía web 2/2

Paso 4: Extraemos el contenido que hemos descargado y lo instalamos:

```
sudo tar xzf Python-3.8.7.tgz
```

```
cd Python-3.8.7
```

```

ramon@ramon-rd: /opt/Python-3.8.7
(base) ramon@ramon-rd:/opt/Python-3.8.7$ sudo ./configure --enable-optimizations
checking build system type... x86_64-pc-linux-gnu
checking host system type... x86_64-pc-linux-gnu
checking for python3.8... python3.8
checking for --enable-universalsdk... no
checking for --with-universal-archs... no
checking MACHDEP... "linux"
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes
checking whether gcc accepts -g... yes
checking for gcc option to accept ISO C89... none needed
checking how to run the C preprocessor... gcc -E
checking for grep that handles long lines and -e... /usr/bin/grep
checking for a sed that does not truncate output... /usr/bin/sed
checking for --with-cxx-main=<compiler>... no
checking for g++... no
configure:

By default, distutils will build C++ extension modules with "g++".

```

Figura 3.9: Instalación de Python 3.8.7 - 1/2

```

ramon@ramon-rd: /opt/Python-3.8.7
(base) ramon@ramon-rd:/opt/Python-3.8.7$ sudo make altinstall
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Programs/python.o ./Programs/python.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/acceler.o Parser/acceler.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/grammar1.o Parser/grammar1.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/listnode.o Parser/listnode.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I
./Include -DPy_BUILD_CORE -o Parser/node.o Parser/node.c
gcc -pthread -c -Wno-unused-result -Wsign-compare -DNDEBUG -g -fwrapv -O3 -Wall
  -std=c99 -Wextra -Wno-unused-result -Wno-unused-parameter -Wno-missing-field-
initializers -Werror=implicit-function-declaration -I./Include/internal -I. -I

```

Figura 3.10: Instalación de Python 3.8.7 - 2/2

NOTA: esta última orden puede demorarse varios minutos.

Para comprobar que la instalación se ha realizado correctamente ejecutamos el siguiente comando:

```
python3.8 -V
```

Y obtendremos como salida:

```
Python 3.8.7
```

NOTA: si ejecutáis en vuestra terminal el comando `python -V` no obtendremos como salida la versión que acabamos de instalar. Para solucionarlo, ejecutaremos las siguientes instrucciones para establecer en nuestro sistema por defecto la versión 3.8.7 de Python:

```
echo "alias python=python3.8" >> ~/.bashrc
```

```
source ~/.bashrc
```

Por último, eliminamos el fichero de instalación que descargamos anteriormente, para liberar espacio en el disco. Para ello:

```
cd /opt
```

```
sudo rm -f Python3.8.7.tgz
```

Con esto concluimos la instalación de Python en su versión 3.8.7. A continuación, estudiaremos la herramienta `pip` para gestionar los paquetes que necesitemos utilizar en nuestros programas.

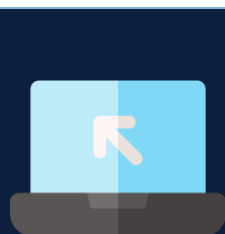
Gestor de paquetes Pip en linux

El gestor de paquetes pip nos permitirá instalar, actualizar y eliminar paquetes que utilizaremos en nuestros programas, como por ejemplo numpy, pandas o seaborn. Para invocar al gestor de paquetes de nuestra versión de python ejecutaremos alguna de las siguientes órdenes.

```
python -m pip install paquete           // Instalar la última versión
python -m pip install paquete==1.1.4   // Instalar una versión específica
python -m pip install paquete >= 1.1.4 // Instalar una versión mínima
python -m pip uninstall [opciones] <paquete> ...
python -m pip uninstall [opciones] -r <requisitos> ...
```

Donde las opciones pueden ser:

```
-r, --requirement <file> // Desinstala todos los paquetes indicados en los
                           requisitos
-y, --yes                 // Desinstala los paquetes indicados sin esperar
                           confirmacion
```



Para más información

Puede consultar la guía de comandos completa de la herramienta pip disponible en el siguiente enlace:

<https://pip.pypa.io/en/stable/reference/pip/>

Instalación de Python en Windows 10

A diferencia de los sistemas UNIX, Windows no incluye un sistema que soporte la instalación de Python. Sin embargo, para poder utilizar Python en Windows, los desarrolladores de Cpython han compilado un conjunto de instaladores (paquetes MSI) para poder llevar a cabo la instalación.

Cabe destacar que la versión de Python 3.8 (y futuras versiones) solo son compatibles con las versiones Windows Vista y superiores. Si en sus máquinas disponen de la versión Windows XP deben instalar la versión 3.4.

Paso 1: Accedemos a la página oficial de Python, a la sección de descargas para Windows, y descargamos la versión 3.8.7. En mi caso, mi máquina es de 64 bits.

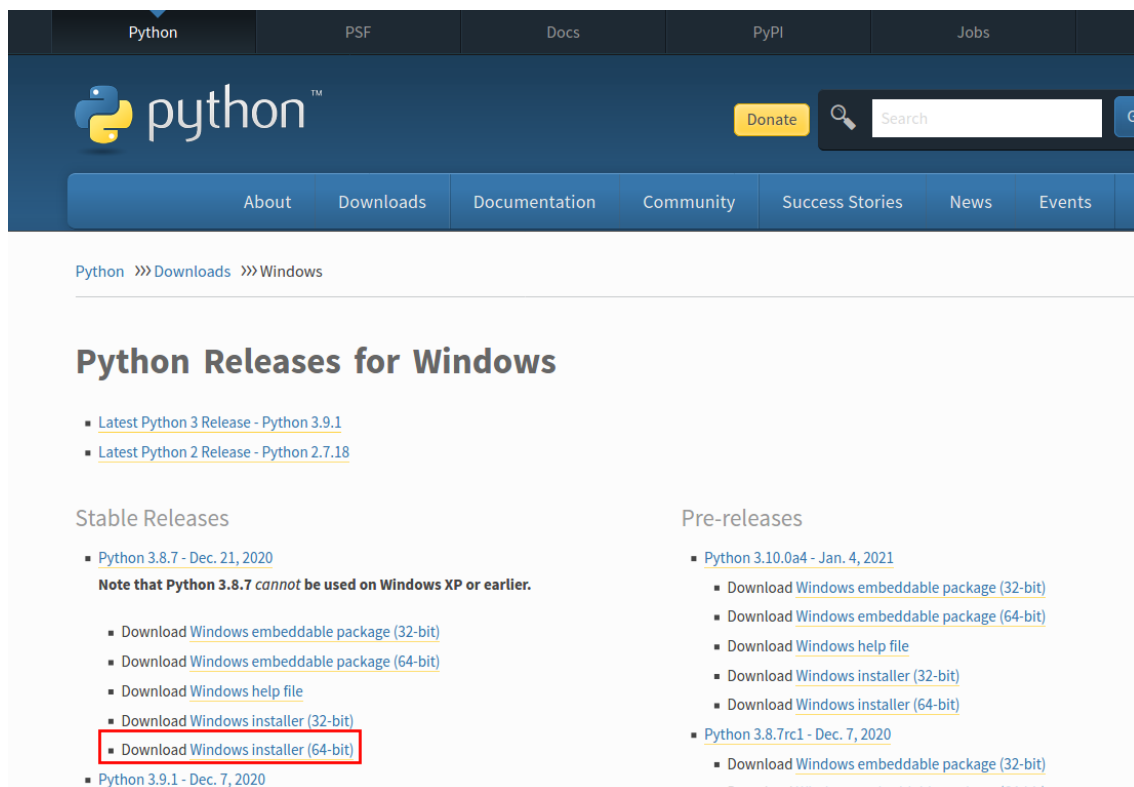


Figura 3.11: Descarga de Python 3.8.7 para Windows 10

Paso 2: Ejecutamos el fichero descargado y comenzamos con la instalación.



Figura 3.13: Instalación de Python 3.8.7 en Windows 10 - 1/3

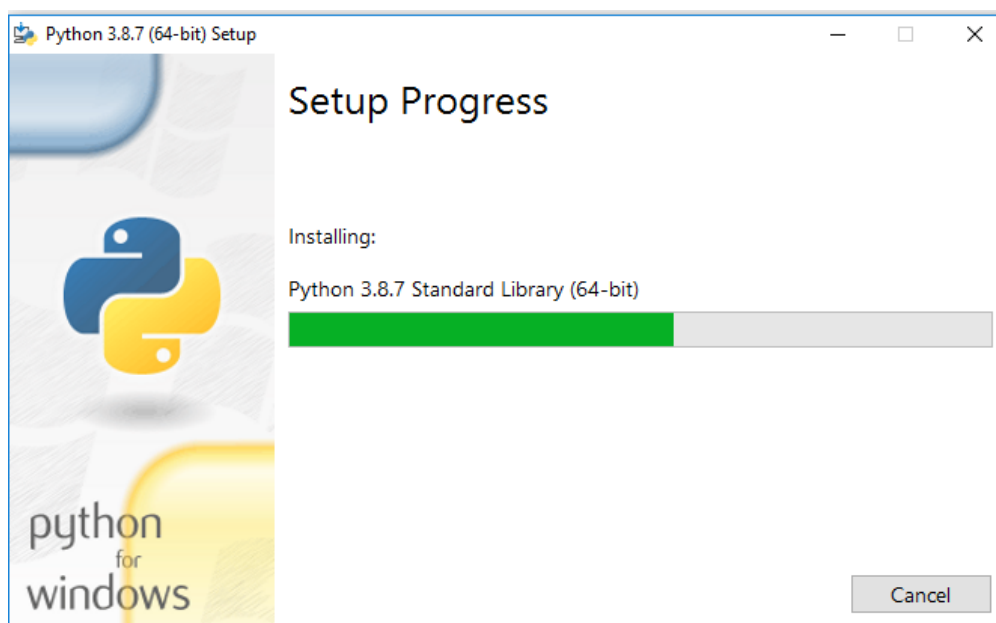


Figura 3.12: Instalación de Python 3.8.7 en Windows 10 - 2/3

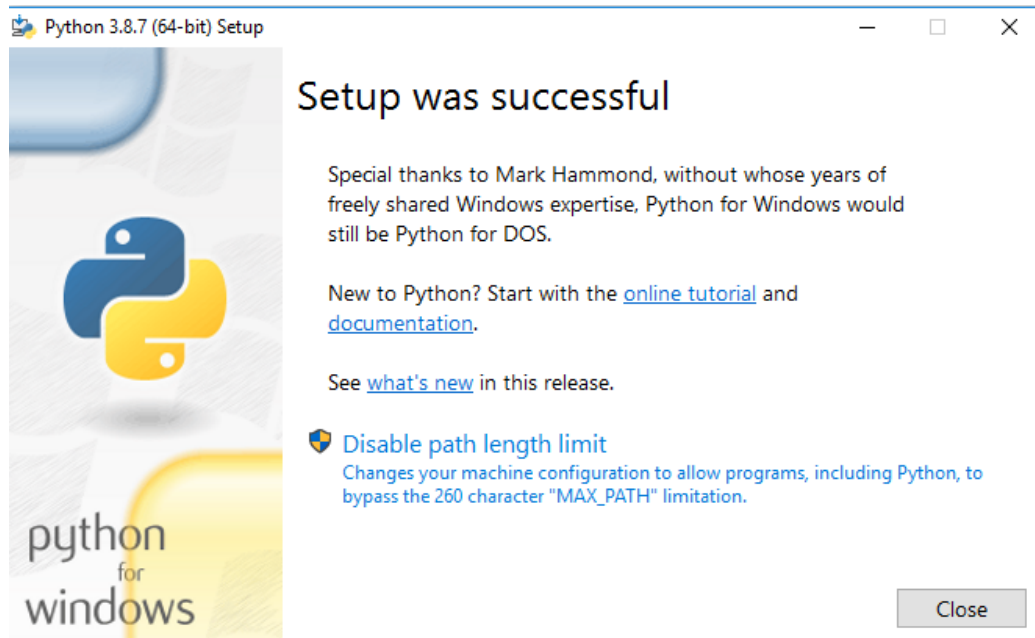


Figura 3.14: Instalación de Python 3.8.7 en Windows 10 - 3/3

Paso 3: Para comprobar que la instalación ha sido realizada con éxito, escribimos en una terminal la orden: `python --version`

Gestor de paquetes Pip en Windows

Al igual que sucede en el sistema operativo linux, Windows también dispone del gestor de paquetes pip. Este comando nos permitirá, entre otras, instalar, actualizar y eliminar paquetes que serán utilizados en nuestras implementaciones. A continuación, mostramos algunas de las órdenes más utilizadas,:

```
py -m pip install SomePackage          # última versión
```

```
py -m pip install SomePackage==1.0.4   # instala una versión específica
```

```
py -m pip install 'SomePackage>=1.0.4' # versión mínima
```

```
python -m pip uninstall [opciones] <package> ...
```

```
python -m pip uninstall [opciones] -r <requirements file> ...
```

Donde las opciones pueden ser:

```
-r, --requirement <file> // Desinstala todos los paquetes indicados en los  
                           requisitos  
-y, --yes                 // Desinstala los paquetes indicados sin esperar  
                           confirmación
```



Para más información

Puede consultar la guía de comandos completa de la herramienta pip disponible en el siguiente enlace:

<https://pip.pypa.io/en/stable/reference/pip/>

4 PUNTOS CLAVE

En esta lección hemos aprendido:

- |Cuál es la posición estratégica de Python.
- |Cómo determinar la popularidad de un lenguaje de programación.
- |Preparar el entorno de desarrollo e instalar Python en los sistemas operativos Windows y Ubuntu.
- |Gestionar la instalación de paquetes en Python por medio del gestor de paquetes pip.

